7장. 클래스

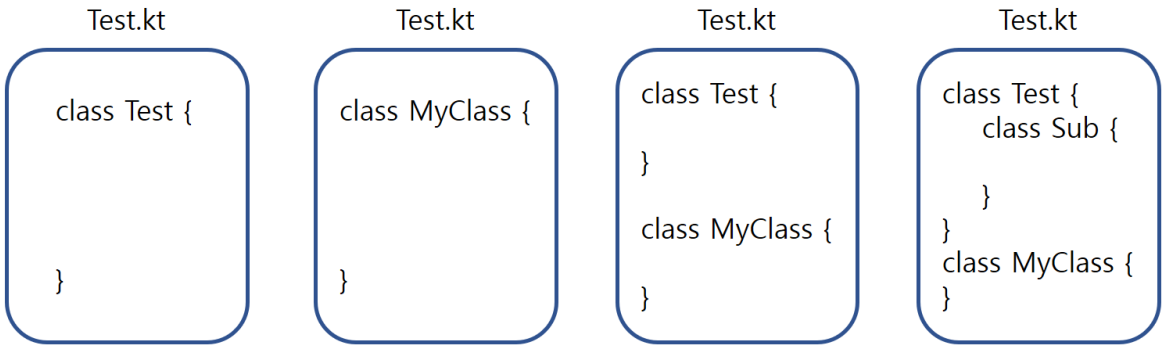
7.1. 클래스 선언 및 생성

7.1.1. 클래스 선언

- class 예약어로 선언
- 클래스 몸체가 없다면 { }는 생략 가능

```
class MyClass { }
```

```
class MyClass
```



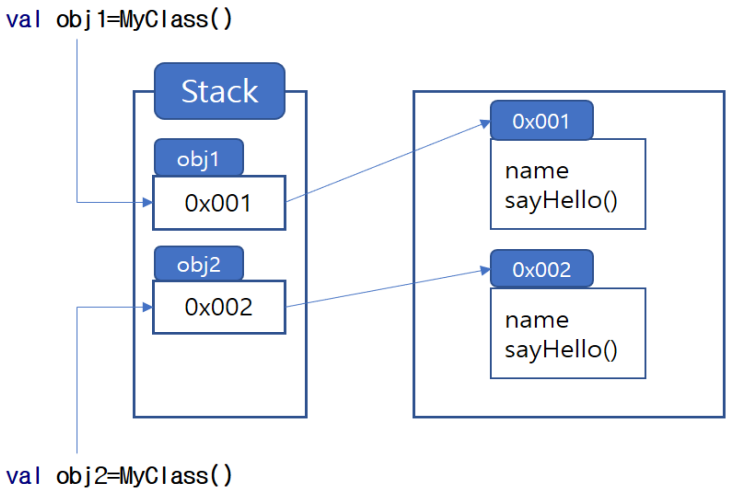
7.1. 클래스 선언 및 생성

```
class MyClass {
    val myVariable=10

    constructor(){ }

    fun myFun(){ }

    class Inner {

    }
}
```

7.1. 클래스 선언 및 생성

7.1.2. 객체 생성

```
class MyClass {
    var name: String="world"

    fun sayHello(){
        println("hello $name")
    }
}

fun main(args: Array<String>) {
    val obj1=MyClass()
    val obj2=MyClass()
}
```



7.2. 생성자

7.2.1. 주 생성자

- 클래스 선언부분에 작성
- 하나의 클래스에 하나의 주 생성자만 정의 가능
- 필수 작성 대상은 아니며 보조 생성자가 있다면 작성 안될수도 있음.

```
class MyClass1 { }

class MyClass2() { }

class MyClass3 constructor() { }

fun main(args: Array<String>) {
    val obj1=MyClass1()
    val obj2=MyClass2()
    val obj3=MyClass3()
}
```

7.2. 생성자

매개변수를 가지는 주생성자

```
class User1 constructor(name: String, age: Int){ }
class User2(name: String, age: Int){ }//.....val user1=User1()//error
val user2=User1("kkang", 33)
val user3=User2("kim", 28)
```

생성자 매개변수 기본값 명시

```
class User3(name: String, age: Int = 0) { }
//.....val user4=User3("kkang", 33)
val user5=User3("kkang")
```

생성자 초기화 블록

```
class User4(name: String, age: Int) { }{ //error

}
```

```
class User4(name: String, age: Int) {
    init {
        println("i am init...")
    }
}
//.....val user6=User4("kkang", 33)
```

7.2. 생성자

생성자 매개변수 값 이용

- 클래스의 초기화 블록, 클래스 프로퍼티에서는 접근이 되지만 클래스에 정의된 함수에서는 사용이 불가능

```
class User5(name: String, age: Int){
    init {
        println("i am init... constructor argument : $name .. $age")
    }
    val upperName=name.toUpperCase()

    fun sayHello(){
        println("hello $name")//error
    }
}
```

- 생성자 내에서 val, var 을 이용해 매개변수를 선언

```
class User6(val name: String, val age: Int){
    val myName=name
    init {
        println("i am init... constructor argument : $name .. ${age}")
    }
    fun sayHello() {
        println("hello $name")
    }
}
```

7.2. 생성자

생성자 매개변수와 클래스 프로퍼티

```
class User7(name: String, age: Int){

    val name: String="kim"

    init {
        println("i am init... constructor argument : $name") //kkang
    }
    fun sayHello() {
        println("hello $name") //kim
    }
}
//.................
val user9=User7("kkang", 33)
user9.sayHello()
```

```
class User8(val name: String, age: Int){
    val name: String="kkang"//error
    val age=10
}
```

7.2. 생성자

7.2.2. 보조 생성자

- 주생성자와 보조생성자로 구분
- 주생성자는 클래스 선언 영역에 작성하는 생성자이며 보조 생성자는 클래스 바디 영역에 constructor 예약어로 선언하는 생성자
- 클래스 선언시 최소한 하나 이상의 생성자는 정의되어 있어야 한다.

```
//컴파일러에 의해 매개변수 없는 주 생성자 자동 추가
class User1 { }

//주생성자만 선언
class User2(name: String) { }

//보조 생성자만 선언
class User3 {
    constructor(name: String){ }
}

fun main(args: Array<String>) {
    val user1=User1()
    val user2=User2("kkang")
//    val user3=User3()//error
    val user4=User3("kkang")
}
```

7.2. 생성자

생성자 오버로딩

```
class User4 {
    constructor(){}
    constructor(name: String){}
    constructor(name: String, age: Int){}
}//.................val user5=User4()
val user6=User4("kkang")
val user7=User4("kkang", 10)
```

보조 생성자 초기화 블록

```
class User5 {
    init {
        println("i am init block....")
    }
    constructor(){
        println("i am constructor....")
    }
}
//.................
val user8=User5()
```

7.2. 생성자

보조생성자의 매개변수 사용

```
class User6 {
    init {
        println("i am init block....$name")//error
    }
    constructor(name: String){
        println("i am constructor....$name")
    }
    fun sayHello(){
        println("hello $name")//error
    }
}
```

```
class User6 {
    constructor(val name: String){//error
        println("i am constructor....$name")
    }
}
```

7.2. 생성자

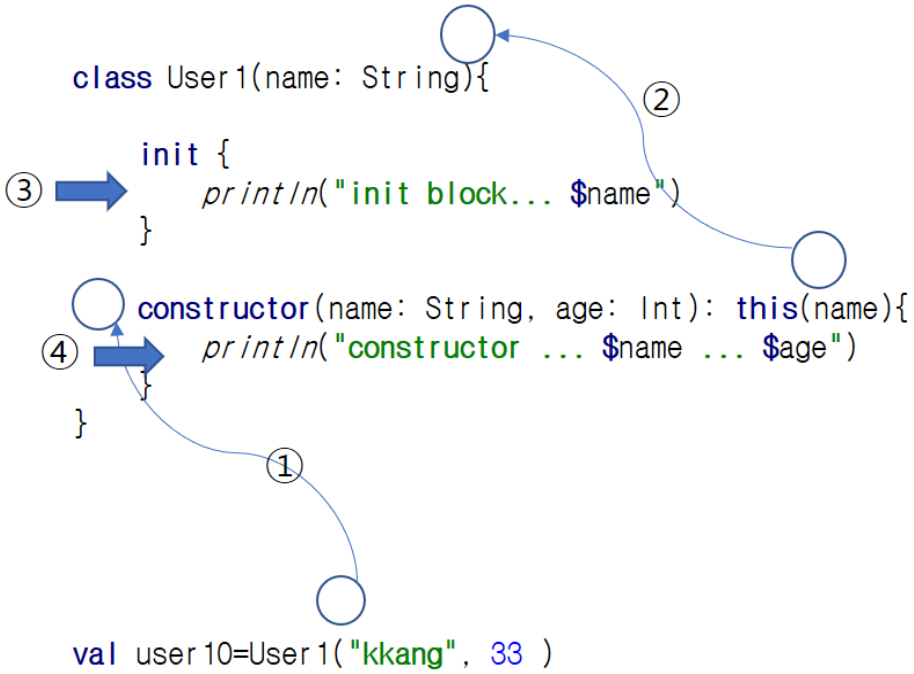
7.2.3. this() 에 의한 생성자 연결

- 주생성자가 선언되어 있다면 보조생성자는 무조건 주 생성자를 같이 호출해 주어야 한다.

```
class User1(name: String){

    constructor(name: String, age: Int){//error

    }
}
```

```
//주생성자와 보조생성자 같이 선언
class User1(name: String){
    init {
        println("init block... $name")
    }
    constructor(name: String, age: Int): this(name){
        println("constructor ... $name ... $age")
    }
}

fun main(args: Array<String>) {
    println("----- 주생성자로 생성한 경우 -------")
    val user1=User1("kkang")
    println("----- 보조생성자로 생성한 경우 -------")
    val user2=User1("kkang", 33)
}
```

7.2. 생성자


class User1(name: String){
init {
println("init block... $name")
}
constructor(name: String, age: Int): this(name){
println("constructor ... $name ... $age")
}
}
val user10=User1("kkang", 33)
①
②
③
④

7.2. 생성자

```
//보조생성자 여러개의 경우
class User2(name: String){
    constructor(name: String, age: Int): this(name){
    }
    constructor(name: String, age: Int, email: String): this(name, age) {
    }
}

fun main(args: Array<String>) {
    //보조생성자 여러개선언된 경우
  val user3=User2("kkang")
    val user4=User2("kkang", 30)
    val user5=User2("kkang", 30, "a@a.com")
}
```